

# *S H I P*

## *SIMPLE HOST INTERSOCKET PROTOCOL*

9 February 1995

*Version 1.0*

### Authors:

*Brian Beattie, Intel Corporation*

*Marty Halvorson, LANL*

*Craig Idler, LANL*

*Jeff Kravitz, IBM*

*Mitch Sukalski, LANL*

*Richard Thomsen, LANL (editor)*

### **Abstract**

This document describes a mechanism for an off-board protocol processor to supply Internet Protocol processing to an attached host. There are new classes of computers and peripherals that are not well architected for onboard Internet Protocol processing, such as massively parallel machines and disk arrays, which provided the impetus for this proposal. It is designed to provide a simple method of transferring data between an application running on the host and the protocol processor, and to be much simpler to implement than the Internet Protocol. In addition, this protocol may help stimulate a growing area of research where I/O protocol processing is not shared with the "main" CPU. At Los Alamos National Laboratory (LANL), the initial intent is to implement this on machines that are physically separate, but connected via some network medium. This particular protocol in no way hinders implementing it using a dedicated I/O processor or process on the same host, nor is it dependent on any particular physical network.



## **Introduction**

SHIP is designed to be used between a host computer and another computer acting as an off-board protocol processor. The protocol processor receives control and data messages from the host and converts the data to the Internet family of protocols for transmission over a network. Likewise, it receives control and data messages using the Internet family of protocols from the network, and sends the data to the host. This transaction relieves the host from the burden of protocol processing. It is especially useful for disk arrays and massively parallel machines, which are not suited for protocol processing.

The code that runs on the host implements a socket library, which looks to the application as standard Berkeley socket interface calls. It packages up these calls in the following defined formats, and sends them to the Protocol Processor. In this manner, applications on the host see the standard system call interface and do not need to be modified to run on the host. The Protocol Processor itself handles all the socket and protocol work.

Most socket calls package up the necessary parameters and send them to the Protocol Processor to be executed. Some calls, such as the read and write commands, are a bit more complicated. The SELECT call is also more complicated, as applications may perform a select on some sockets as well as some local devices, such as a terminal. Select is implemented as a status call within the socket library, where the host can poll the Protocol Processor to see the status of the sockets of interest, and can block or not, as desired. It can also cancel any blocking status call.

The socket calls that are implemented are as follows:

- Socket Creation
- Bind
- Connect
- Listen
- Accept
- Accept (non-blocking)
- Write
- Read (blocking and non-blocking)
- Close
- Shutdown
- Getsockopt
- Setsockopt
- Status
- Status (non-blocking)
- Cancel Status

## **Definitions**

### **General Definitions**

### **Definitions for Protocol Processor**

These definitions are used in the descriptions of the actions of the PP, and not definitions of fields in the protocol packets themselves.

Socket Control Block (SCB):

The internal PP control block used to contain all information about a particular socket.

## **Protocol Format**

The format of the SHIP header consists of the following fields. All fields are 32 bits unless stated. All information is stored in network byte order.

- Host ID: ID of host making request. This is used when more than one host is connected to a protocol processor. This field is 8 bits wide.
- Status: On return, status of protocol. If there was a protocol error, such as an invalid field, it is indicated here. If the protocol was valid, but the command was not performed, then it indicates that a command error occurred. This field is 8 bits wide.
- Function Code: The command to be performed. This field is 16 bits wide, and is divided up into a Major Function Code (upper 8 bits), and a Minor Function Code (lower 8 bits). At this point, the Major Function Code is zero for these operations, and all others are reserved.
- Request ID: A unique ID for this request. This must be monotonically increasing (i.e., each succeeding one is larger than the last, except for wrap around). They are not required to be sequential.
- Host Socket ID 1: First ID word used by the host to identify this socket. It is not interpreted by the Protocol Processor, but is returned so the host can use it to index into its tables.
- Host Socket ID 2: Second ID word used by the host to identify this socket. It is not interpreted by the Protocol Processor, but is returned so the host can use it to index into its tables.
- PP Socket ID 1: First ID word used by the Protocol Processor to identify this socket. It is not interpreted by the host, but is returned so the Protocol Processor can use it to index into its tables.
- PP Socket ID 2: Second ID word used by the Protocol Processor to identify this socket. It is not interpreted by the host, but is returned so the Protocol Processor can use it to index into its tables.
- Extension Length: The number of bytes of additional information following this header.
- Data Length: The number of bytes of data following this header.
- Command Status: In the response message, the status of the command. This tells the host what the result of the command was. For example, if a Create Socket was requested, but no more sockets were available, the Status field would indicate command error, and the Command Status field would indicate no more sockets. This field is reserved on the commands.
- Parameter: This contains a parameter for the command or response.

The data area consists of the data needed for the socket calls in network byte order.

All addresses, expressed by Server Name, are passed using the sockaddr structure. This structure is documented in an appendix.

For HIPPI links, the SHIP header is put in the HIPPI-FP D1 area. The data, if any, is put in the D2 area. For non-HIPPI links, the header and data together make up the payload of the link protocol.

Reserved fields must be sent as zero.

### **Initialization Commands**

These commands tell the Protocol Processor that the host wishes to initialize the connection or a socket.

## Initialize Command

This command tells the Protocol Processor to initialize the socket processing for this particular host. Any currently open sockets are closed, and the Protocol Processor assumes that the host has restarted.

The options field gives overall SHIP processing options for that host. Currently defined options are as follows:

None Yet Defined

Header :

Host ID	Reserved	Function Code = Initialize
	Request_id	
	Reserved	
	Reserved	
	Reserved	
	Reserved	
	Extension Length = 0	
	Data Length = 0	
	Reserved	
	Options	

## PP Actions:

Close any existing sessions with peer nodes (discarding any buffered data).  
Deallocate all outstanding resources (e.g. buffers, data, SCB's).  
Reset all internal variables, switches, etc. to initial state.  
Return SHIP Initialize response to host immediately.



### Initialize Response

This response tells the host when the Protocol Processor has been initialized for that host.

The options field gives overall SHIP processing options for that host. Currently defined options are as follows:

None Yet Defined

Header :

Host ID	Status	Function Code = Initialize
Request_id		
Reserved		
Reserved		
Reserved		
Reserved		
Extension Length = 0		
Data Length = 0		
Reserved		
Options		

### Possible Status Values:

SHIP_STAT_UNREC_HOST	Unrecognized Host ID
SHIP_STAT_UNREC_FCN	Unrecognized function code

### Socket Creation Command

This command tells the Protocol Processor to set up a socket for communication with a remote system, specifying the protocol to be used.

The options field gives options for that socket. Currently defined options are as follows:

PP must read and write exact amount of data specified.

Header :

Host ID	Reserved	Function Code = Create Socket
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
Reserved - must be zero		
Reserved - must be zero		
Extension Length = 16		
Data Length = 0		
Reserved		
Reserved		
Domain		
Type		
Protocol		
Options		

### PP Actions:

Allocate an SCB for a new socket and initialize it.

If allocation is not possible (i.e. all SCB's are in use)

Return an error in the Create response SHIP packet.

Otherwise,

Save any specified option values in the new SCB.

Return a Create response SHIP packet containing the PP Socket ID.

### Socket Creation Response

This response tells the host if the Protocol Processor was successful in setting up a socket.

The options field gives options for that socket. Currently defined options are as follows:

None yet defined.

Header :

Host ID	Status	Function Code = Create Socket
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = 16		
Data Length = 0		
Command Status		
Reserved		
Maximum reads outstanding		
Maximum writes outstanding		
Maximum read size allowed		
Maximum write size allowed		

### Possible Status Values:

SHIP\_STAT\_MUSTINIT  
SHIP\_STAT\_UNREC\_HOST  
SHIP\_STAT\_UNREC\_FCN  
SHIP\_STAT\_INVLENGTH  
SHIP\_STAT\_CMDERROR

Initialization not done  
Unrecognized Host ID  
Unrecognized function code  
Data length was not valid  
Command error on creation of socket

## **Socket Commands**

These commands tell the Protocol Processor that the host wishes to perform some action on the socket itself.

## Bind Command

This command binds a name (Internet address) with the local socket set up by the Protocol Processor by the socket create command. This name will be used by remote hosts to talk with this socket.

If the server name is zero (unspecified), the PP will fill in the default host name. If the port number is zero (unspecified), it will use an unused port.

Header :

Host ID	Reserved	Function Code = Bind
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = sizeof (Server Name)		
Data Length = 0		
Reserved		
Reserved		
Server Name		

## PP Actions:

Save the name specified by the Bind SHIP packet in the SCB.  
Return Bind response SHIP packet to host immediately.

## Bind Response

This response tells the host if the Protocol Processor was successful in binding the name to the socket, and returns the address and port that was assigned.

Header :

Host ID	Status	Function Code = Bind
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = sizeof (Server Name)		
Data Length = 0		
Command Status		
Reserved		
Server Name		

## Possible Status Values:

SHIP_STAT_MUSTINIT	Initialization not done
SHIP_STAT_UNREC_HOST	Unrecognized Host ID
SHIP_STAT_UNREC_FCN	Unrecognized function code
SHIP_STAT_INVLENGTH	Data length was not valid
SHIP_STAT_INVPPSOC	Invalid PP Socket ID words
SHIP_STAT_CMDERROR	Command error on creation of socket

## Connect Command

This command tells the Protocol Processor to connect this socket to the specified socket on the remote host.

Header :

Host ID	Reserved	Function Code = Connect
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = sizeof (Server Name)		
Data Length = 0		
Reserved		
Reserved		
Server Name		

## PP Actions:

Attempt to make a network connection with the peer node whose name is specified in the Connect SHIP packet.

If the connection is acknowledged by the peer

Set the SCB to a state that allows data transfer.

Return a Connect response SHIP packet indicating success.

If a timeout occurs before an acknowledgment is received from the peer

Return a Connect response SHIP packet indicating an Error.

If the peer explicitly rejects the connection attempt

Return a Connect response SHIP packet indicating an Error.

## Connect Response

This response tells the host if the connection was successful.

Header :

Host ID	Status	Function Code = Connect
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = 0		
Data Length = 0		
Command Status		
Reserved		

## Possible Status Values:

SHIP_STAT_MUSTINIT	Initialization not done
SHIP_STAT_UNREC_HOST	Unrecognized Host ID
SHIP_STAT_UNREC_FCN	Unrecognized function code
SHIP_STAT_INVLENGTH	Data length was not valid
SHIP_STAT_INVPPSOC	Invalid PP Socket ID words
SHIP_STAT_CMDERROR	Command error on connect



## Listen Command

This command tells the Protocol Processor to listen for incoming requests to the socket, and the maximum number of outstanding connections that may be queued.

Header :

Host ID	Reserved	Function Code = Listen
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = 0		
Data Length = 0		
Reserved		
Maximum outstanding connections		

## PP Actions:

Set the SCB into a state that allows connection attempts from other nodes.

Allow for a specified number of queued connection attempts.

If the listen succeeds, i.e. state change was acceptable

Return a Listen response SHIP packet indicating success.

If the listen fails, i.e. illegal state change

Return a Listen response SHIP packet indicating an Error.

Note: The Listen response does NOT indicate that a connection attempt was received.

## Listen Response

This response tells the host the status of the listen command.

Header :

Host ID	Status	Function Code = Listen
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = 0		
Data Length = 0		
Command Status		
Reserved		

## Possible Status Values:

SHIP_STAT_MUSTINIT	Initialization not done
SHIP_STAT_UNREC_HOST	Unrecognized Host ID
SHIP_STAT_UNREC_FCN	Unrecognized function code
SHIP_STAT_INVLENGTH	Data length was not valid
SHIP_STAT_INVPPSOC	Invalid PP Socket ID words
SHIP_STAT_CMDERROR	Command error on listen

## Accept Command

This command tells the Protocol Processor to accept connections on this socket, and allows a connection from the remote host. It will not respond until a connection is accepted.

Header :

Host ID	Reserved	Function Code = Accept
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = 0		
Data Length = 0		
Reserved		
Reserved		

## PP Actions:

Set the SCB into a state that causes the next (or current) connection attempt from a network peer to be acknowledged to the peer.

If a connection attempt is already queued

- Create a new socket for the connection, and initialize its SCB.

- Acknowledge that connection to the peer.

- Set the new SCB to a state that allows data transfer.

- Return a successful Accept response to the host, containing the new socket ID.

Otherwise,

- Mark the SCB indicating that an Accept is pending

- Delay the Accept response until a connection attempt is received from the network, and acknowledged.

- Later, when a connection attempt is received from the network

  - Create a new socket for the connection, and initialize its SCB.

  - Acknowledge that connection to the peer

  - Set the new SCB to a state that allows data transfer.

  - Return a successful Accept response to the host, containing the new socket ID.

## Accept Response

This response tells the host that an incoming connection has been established.

The PP Socket ID word indicates the new socket that was created by the accept, not the one originally specified by the ACCEPT command.

Header :

Host ID	Status	Function Code = Accept
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = sizeof (Server Name)		
Data Length = 0		
Command Status		
Reserved		
Server Name		

## Possible Status Values:

SHIP_STAT_MUSTINIT	Initialization not done
SHIP_STAT_UNREC_HOST	Unrecognized Host ID
SHIP_STAT_UNREC_FCN	Unrecognized function code
SHIP_STAT_INVLENGTH	Data length was not valid
SHIP_STAT_INVPPSOC	Invalid PP Socket ID words
SHIP_STAT_CMDERROR	Command error on accept
SHIP_STAT_CANCELED	Command was canceled

### Accept Noblock Command

This command tells the Protocol Processor to accept connections on this socket, and allows a connection from the remote host. It will return immediately with a response.

Header :

Host ID	Reserved	Function Code = Accept Noblock
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = 0		
Data Length = 0		
Reserved		
Reserved		

### PP Actions:

If a connection attempt from a peer node is pending

- Create a new socket for the connection, and initialize its SCB.

- Acknowledge that connection to the peer

- Set the new SCB to a state that allows data transfer.

- Return a successful Accept response to the host, containing the new socket ID.

Otherwise,

- Return an error response to the host.

### Accept Noblock Response

This response tells the host if an incoming connection has been established. If it would have blocked, waiting for a connection, it will return with extension length of zero and no server name.

The PP Socket ID word indicates the new socket that was created by the accept, not the one originally specified by the ACCEPT command.

Header :

Host ID	Status	Function Code = Accept Noblock
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = sizeof (Server Name)		
Data Length = 0		
Command Status		
Reserved		
Server Name		

### Possible Status Values:

SHIP_STAT_MUSTINIT	Initialization not done
SHIP_STAT_UNREC_HOST	Unrecognized Host ID
SHIP_STAT_UNREC_FCN	Unrecognized function code
SHIP_STAT_INVLENGTH	Data length was not valid
SHIP_STAT_INVPPSOC	Invalid PP Socket ID words
SHIP_STAT_CMDERROR	Command error on accept

## Close Command

This command tells the Protocol Processor that the socket is to be closed. The connection is closed in both directions, and the socket is closed.

Header :

Host ID	Reserved	Function Code = Close
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = 0		
Data Length = 0		
Reserved		
Reserved		

## PP Actions:

Discard any data enqueued at this SCB in the PP for transmission, and any data enqueued at this SCB awaiting reception by the host.

Send an end-of-data indication to the peer node.

Do not wait for an acknowledgment to the end-of-data indication.

Mark the SCB as closed, preventing any further SHIP functions to be issued to this socket by the host (until the socket ID is subsequently re-allocated by Create).

Any pending operations awaiting responses (e.g. Status, Accept, Read, etc.) will not be completed.

Any data received from the network for this socket should be discarded.

Note: It is prudent to ensure that this socket ID is not reused by Create for some period of time, to reduce the possibility of outstanding data on the network being sent to a new incarnation of this socket ID.

## Close Response

This response tells the host the result of the close.

Header :

Host ID	Status	Function Code = Close
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = 0		
Data Length = 0		
Command Status		
Reserved		

## Possible Status Values:

SHIP_STAT_MUSTINIT	Initialization not done
SHIP_STAT_UNREC_HOST	Unrecognized Host ID
SHIP_STAT_UNREC_FCN	Unrecognized function code
SHIP_STAT_INVLENGTH	Data length was not valid
SHIP_STAT_INVPPSOC	Invalid PP Socket ID words
SHIP_STAT_CMDERROR	Command error on close



## Shutdown Command

This command tells the Protocol Processor that the connection is being shut down. The value Shutdown How indicates the manner of the shutdown.

Header :

Host ID	Reserved	Function Code = Shutdown
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = 0		
Data Length = 0		
Reserved		
Shutdown How		

The values for the Shutdown How field is as specified in the socket description; 0 for no more input, 1 for no more output, and 2 for neither input nor output.

### PP Actions:

Mark the SCB indicating that no further Reads/Writes will be permitted.

If Writes are no longer permitted

Any data already enqueued in the PP for transmission should be sent to the peer node and an indication of end-of-data will be sent afterward to the peer node.

Delay the response to the host until the end-of-data indication is acknowledged by the peer, or until some timeout condition occurs.

If the end-of-data indication is acknowledged by the peer

Return a successful indication to the host.

If the wait for the acknowledgment times out

Return an error response to the host.

If Reads are no longer permitted

Any previously received network data enqueued in the PP not yet received by the host should be discarded.

## Shutdown Response

This response tells the host the result of the shutdown command.

Header :

Host ID	Status	Function Code = Shutdown
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = 0		
Data Length = 0		
Command Status		
Reserved		

## Possible Status Values:

SHIP_STAT_MUSTINIT	Initialization not done
SHIP_STAT_UNREC_HOST	Unrecognized Host ID
SHIP_STAT_UNREC_FCN	Unrecognized function code
SHIP_STAT_INVLENGTH	Data length was not valid
SHIP_STAT_INVPPSOC	Invalid PP Socket ID words
SHIP_STAT_CMDERROR	Command error on shutdown

### Status Command

This command tells the Protocol Processor to return a response when data has appeared for a socket. A response will not be made until there is data to be read or, if writes were prevented, until writes are permitted on this socket.

Header :

Host ID	Reserved	Function Code = Status
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = 0		
Data Length = 0		
Reserved		
Reserved		

### PP Actions:

If there is previously received network data enqueued at the SCB

Return a Status Response SHIP packet, containing the Status of the socket

Otherwise,

Mark the SCB as having a Status command pending, indicating that when any network data arrives for the socket, a Status Response SHIP packet is to be sent back to the host.

Later, when any data is received from the network

Return a successful Status response to the host.

Mark the SCB as no longer having a Status command pending,

## Status Response

This response tells the host the status of the socket.

The Bytes Ready for Transfer tells the host how much data the Protocol Processor currently has buffered for the host on this socket.

Header :

Host ID	Status	Function Code = Status
	Request_id	
	First Host Socket ID word	
	Second Host Socket ID word	
	First PP Socket ID word	
	Second PP Socket ID word	
	Extension Length = 12	
	Data Length = 0	
	Command Status	
	Reserved	
	Bytes Ready for Transfer	
	Current State	
	Number of reads queued	

## Possible Status Values:

SHIP_STAT_MUSTINIT	Initialization not done
SHIP_STAT_UNREC_HOST	Unrecognized Host ID
SHIP_STAT_UNREC_FCN	Unrecognized function code
SHIP_STAT_INVLENGTH	Data length was not valid
SHIP_STAT_INVPPSOC	Invalid PP Socket ID words
SHIP_STAT_CMDERROR	Command error on status
SHIP_STAT_CANCELED	Command was canceled

### Status Noblock Command

This command tells the Protocol Processor to return a response when data has appeared for a socket. The response will be returned immediately.

Header :

Host ID	Reserved	Function Code = Status Noblock
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = 0		
Data Length = 0		
Reserved		
Reserved		

### PP Actions:

Immediately return a Status Response SHIP packet, containing the Status of the socket

## Status Noblock Response

This response tells the host the status of the socket.

The Bytes Ready for Transfer tells the host how much data the Protocol Processor currently has buffered for the host on this socket.

Header :

Host ID	Status	Function Code = Status Noblock
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = 12		
Data Length = 0		
Command Status		
Reserved		
Bytes Ready for Transfer		
Current State		
Number of reads queued		

## Possible Status Values:

SHIP_STAT_MUSTINIT	Initialization not done
SHIP_STAT_UNREC_HOST	Unrecognized Host ID
SHIP_STAT_UNREC_FCN	Unrecognized function code
SHIP_STAT_INVLENGTH	Data length was not valid
SHIP_STAT_INVPPSOC	Invalid PP Socket ID words
SHIP_STAT_CMDERROR	Command error on status

## Cancel Command

This command tells the Protocol Processor to cancel any previous blocking command (such as ACCEPT, READ, WRITE, or STATUS, etc.), and not to send a response for it (a WRITE can be canceled only if the PP is holding the response until there is some buffer space available).

Header :

Host ID	Reserved	Function Code = Cancel
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = 0		
Data Length = 0		
Reserved		
Reserved		

## PP Actions:

If the Cancel is for Status Block, or Accept Block commands, cancel them, returning Status or Accept Responses indicating that they were canceled.

If the Cancel is for Read commands and some are pending, cancel all of them, returning a Read Response only for the last one, indicating that it was canceled.

If none of the above commands are pending, return a Cancel Response.

## Cancel Response

This response is only returned if no blocking request was canceled when the CANCEL was issued.

If there is no command to be canceled, then this response will indicate why the cancel command was not able to complete. If it does cancel a blocking command, the response will be the normal response to that command with a status of canceled.

Header :

Host ID	Status	Function Code = Cancel
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = 0		
Data Length = 0		
Command Status		
Reserved		

## Possible Status Values:

SHIP_STAT_MUSTINIT	Initialization not done
SHIP_STAT_UNREC_HOST	Unrecognized Host ID
SHIP_STAT_UNREC_FCN	Unrecognized function code
SHIP_STAT_INVLENGTH	Data length was not valid
SHIP_STAT_INVPARAM	Invalid parameter to call
SHIP_STAT_INVPPSOC	Invalid PP Socket ID words



## Getsockopt Command

This command allows the host to get some options from the socket.

Header :

Host ID	Reserved	Function Code = Get Socket Option
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = 8		
Data Length = 0		
Reserved		
Reserved		
Level		
Socket Option Name		

## PP Actions:

Parse the SHIP packet to determine the identification of the option data being requested

If the Parse is successful

Return the specified information to the host, immediately.

Otherwise,

Return a Getopt response packet indicating success.

## Getsockopt Response

This response tells the host the status of the Getsockopt command.

The data contains the current data for the command. The value, if any, is in Socket Option Value, and the length is either zero or four (bytes).

Header :

Host ID	Status	Function Code = Get Socket Option
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = 16		
Data Length = 0		
Command Status		
Reserved		
Level		
Socket Option Name		
Socket Option Length		
Socket Option Value		

## Possible Status Values:

SHIP_STAT_MUSTINIT	Initialization not done
SHIP_STAT_UNREC_HOST	Unrecognized Host ID
SHIP_STAT_UNREC_FCN	Unrecognized function code
SHIP_STAT_INVLENGTH	Data length was not valid
SHIP_STAT_INVPPSOC	Invalid PP Socket ID words
SHIP_STAT_CMDERROR	Command error on get socket option

## Setsockopt Command

This command allows the host to set some options for the socket.

The value, if any, is in Socket Option Value, and the length is either zero or four.

Header :

Host ID	Reserved	Function Code = Set Socket Option
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = 16		
Data Length = 0		
Reserved		
Reserved		
Level		
Socket Option Name		
Socket Option Length		
Socket Option Value		

## PP Actions:

Parse the SHIP packet to determine which option is being set.

If the Parse is successful

- Store the necessary option values in the SCB.

- Return a Setopt response packet indicating success.

Otherwise,

- Return a Setopt response packet indicating error.

## Setsockopt Response

This response tells the host the status of the Setsockopt command.

Header :

Host ID	Status	Function Code = Set Socket Option
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = 0		
Data Length = 0		
Command Status		
Reserved		

## Possible Status Values:

SHIP_STAT_MUSTINIT	Initialization not done
SHIP_STAT_UNREC_HOST	Unrecognized Host ID
SHIP_STAT_UNREC_FCN	Unrecognized function code
SHIP_STAT_INVLENGTH	Data length was not valid
SHIP_STAT_INVPPSOC	Invalid PP Socket ID words
SHIP_STAT_CMDERROR	Command error on set socket option

### **Data Transfer Commands**

These commands tell the Protocol Processor that the host wishes to transfer data transfer between them.

## Write Command

This command tells the Protocol Processor that the host wants to write data, and the data is included in the command. Block numbers must be a number starting at 1 with the first write, and incrementing by one for each subsequent write. This allows the PP to detect if a write is missing.

If a write is missing or in error, all subsequent writes are ignored by the PP (which returns a status code of SHIP\_STAT\_SEQUENCE) until the missing or erroneous write is redone.

The server name tells the destination to which this data is to be sent. It is only present if there is not a connection already in process. Otherwise, Extension Length is zero.

Header :

Host ID	Reserved	Function Code = Write
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = sizeof (Server Name)		
Data Length = 0		
Reserved		
Block Number		
Server Name		

Data :

Data being Transferred
------------------------

## PP Actions:

If this command exceeds the maximum number of outstanding Write commands, discard the command with no response.

If the data length specified in this command exceeds the maximum supported for this PP, discard the data, returning a response indicating the error, and ignore all subsequent Write commands with larger Request ID numbers until a Write command is issued with the same Request ID as the failed one and a legal length.

Otherwise,

Enqueue the received data for transmission to the peer node.

If there is sufficient buffer space to handle the additional data, return a Write response SHIP packet, indicating that the Write was successful.

Note: All Write Responses contain the amount of free buffer space for subsequent Writes.

If there is no free buffer space available after the current Write, then the response must be delayed until space becomes available. Therefore the response should be enqueued at the SCB.

Otherwise,

Return an unsuccessful Write response to the host. indicating NO ROOM available.

## Write Response

This response tells the host if the transfer was successful or not.

Header :

Host ID	Status	Function Code = Write
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = 0		
Data Length = 0		
Command Status		
Buffer Space Left		

## Possible Status Values:

SHIP_STAT_MUSTINIT	Initialization not done
SHIP_STAT_UNREC_HOST	Unrecognized Host ID
SHIP_STAT_UNREC_FCN	Unrecognized function code
SHIP_STAT_INVLENGTH	Data length was not valid
SHIP_STAT_INVPPSOC	Invalid PP Socket ID words
SHIP_STAT_CMDERROR	Command error on write
SHIP_STAT_SEQUENCE	Write out of sequence
SHIP_STAT_NOROOM	No room for write



## Read Command

This command tells the Protocol Processor that the host wants to read data, and the data should be sent immediately. The data transfer size tells the Protocol Processor the total amount of data that the host wishes to read. If there is no data, the PP waits until there is data before returning the response.

The Request ID Ack will tell the PP that all read requests up to but not including this request ID have been received, and the data may be discarded.

Header :

Host ID	Reserved	Function Code = Read
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = 0		
Data Length = 0		
Request ID Ack		
Data Transfer Size		

## PP Actions:

If this command exceeds the maximum number of outstanding Read commands, discard the command with no response.

If the Request Id field is the same as the one in the previous Read command, return the same data that was returned by the previous Read command.

Using the Request ID ACK field in this command free the all data awaiting acknowledgment from previous Read commands (i.e. ones with lower Request ID fields).

If there is previously received network data enqueued at the socket

Immediately return a Read response SHIP packet containing the data requested or the data available, whichever is smaller, unless the SCB was created with the EXACT DATA SIZE option. If so, and there is less data enqueued than was requested, continue as if no data was enqueued, below.

Enqueue the data buffer at the SCB, and mark the SCB to indicate that Read Data is awaiting acknowledgment.

Otherwise

Mark the SCB as having a Read Block response pending, indicating that when any network data arrives for the socket, a Read response SHIP packet containing the data, is to be sent back to the host

Later, when sufficient data is received from the network (sufficient being defined to mean any amount, if the socket does not have the EXACT DATA SIZE option set, or the amount requested if the option is set).

Return a Read response to the host, containing the data.

Enqueue the data buffer at the SCB, and mark the SCB to indicate that Read Data is awaiting acknowledgment.

Mark the SCB as no longer having a Read response pending,

Note: All Read Responses contain the amount of data enqueued for subsequent Reads.

## Read Response

This response gives the data to the host.

The server name tells the source from which this data was received. It is only present if there is not a connection already in process. Otherwise, Extension Length is zero.

Head

Host ID	Status	Function Code = Read
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = sizeof (Server Name)		
Data Length = 0		
Command Status		
Reserved		
Server Name		

Data:

Data being Transferred
------------------------

## Possible Status Values:

SHIP_STAT_MUSTINIT	Initialization not done
SHIP_STAT_UNREC_HOST	Unrecognized Host ID
SHIP_STAT_UNREC_FCN	Unrecognized function code
SHIP_STAT_INVLENGTH	Data length was not valid
SHIP_STAT_INVPPSOC	Invalid PP Socket ID words
SHIP_STAT_CMDERROR	Command error on read
SHIP_STAT_CANCELED	Command was canceled

## Read Noblock Command

This command tells the Protocol Processor that the host wants to read data, and the data should be sent immediately. The data transfer size tells the Protocol Processor the total amount of data that the host wishes to read. If there is no data, the PP returns the response indicating no data available.

The Request ID Ack will tell the PP that all read requests up to but not including this request ID have been received, and the data may be discarded.

Header :

Host ID	Reserved	Function Code = Read Noblock
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = 0		
Data Length = 0		
Request ID Ack		
Data Transfer Size		

## PP Actions:

If this command exceeds the maximum number of outstanding Read commands, discard the command with no response.

If the Request Id field is the same as the one in the previous Read command, return the same data that was returned by the previous Read command.

Using the Request ID ACK field in this command free the all data awaiting acknowledgment from previous Read commands (i.e. ones with lower Request ID fields).

If there is previously received network data enqueued at the socket

Immediately return a Read response SHIP packet containing the data requested or the data available, whichever is smaller, unless the SCB was created with the EXACT DATA SIZE option. If so, and there is less data enqueued than was requested, continue operation as if a Read (block) was issued.

Enqueue the data buffer at the SCB, and mark the SCB to indicate that Read Data is awaiting acknowledgment.

Otherwise

Immediately return a Read Response indicating no data available.

Note: All Read Responses contain the amount of data enqueued for subsequent Reads.

## Read Noblock Response

This response gives the data to the host. If there is no data, the data length is set to zero.

The server name tells the source from which this data was received. It is only present if there is not a connection already in process. Otherwise, Extension Length is zero.

Header :

Host ID	Status	Function Code = Read Noblock
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = sizeof (Server Name)		
Data Length = 0		
Command Status		
Reserved		
Server Name		

Data :

Data being Transferred
------------------------

## Possible Status Values:

SHIP_STAT_MUSTINIT	Initialization not done
SHIP_STAT_UNREC_HOST	Unrecognized Host ID
SHIP_STAT_UNREC_FCN	Unrecognized function code
SHIP_STAT_INVLENGTH	Data length was not valid
SHIP_STAT_INVPPSOC	Invalid PP Socket ID words
SHIP_STAT_CMDERROR	Command error on read

## Read Redirect Command

This command tells the Protocol Processor that the host wants to read data, and the data should be sent immediately. However, the data is to be sent to another device. The data transfer size tells the Protocol Processor the total amount of data that the host wishes to read. If there is no data, the PP waits until there is data before returning the response.

The Request ID Ack will tell the PP that all read requests up to but not including this request ID have been received, and the data may be discarded.

Header :

Host ID	Reserved	Function Code = Read Redirect
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = 0		
Data Length = 0		
Request ID Ack		
Data Transfer Size		

Data :

Address information and header to be used
---

## PP Actions:

If this command exceeds the maximum number of outstanding Read commands, discard the command with no response.

If the Request Id field is the same as the one in the previous Read command, return the same data that was returned by the previous Read command.

Using the Request ID ACK field in this command free the all data awaiting acknowledgment from previous Read commands (i.e. ones with lower Request ID fields).

If there is previously received network data enqueued at the socket

Immediately return a Read Redirect response SHIP packet to the requester, and a Data Packet to the redirected host, containing the data requested or the data available, whichever is smaller, unless the SCB was created with the EXACT DATA SIZE option. If so, and there is less data enqueued than was requested, continue as if no data was enqueued, below.

Enqueue the data buffer at the SCB, and mark the SCB to indicate that Read Data is awaiting acknowledgment.

Otherwise

Mark the SCB as having a Read Redirect response pending, indicating that when any network data arrives for the socket, a Read Redirect response SHIP packet containing the data, is to be sent back to the host

Later, when sufficient data is received from the network (sufficient being defined to mean any amount, if the socket does not have the EXACT DATA SIZE option set, or the amount requested if the option is set).

Return a Read Redirect response SHIP packet to the requester, and a Data Packet to the redirected host, containing the data.

Enqueue the data buffer at the SCB, and mark the SCB to indicate that Read Data is awaiting acknowledgment.

Mark the SCB as no longer having a Read Redirect response pending.

Note: All Read Responses contain the amount of data enqueued for subsequent Reads.

## Read Redirect Response

This response gives the data to the destination as specified by the READ REDIRECT command.

The server name tells the source from which this data was received. It is only present if there is not a connection already in process. Otherwise, Extension Length is zero.

Header :

Host ID	Status	Function Code = Read Redirect
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = sizeof (Server Name)		
Data Length = 0		
Command Status		
Reserved		
Server Name		

Data :

Header specified from command
Data being transferred

## Possible Status Values:

SHIP_STAT_MUSTINIT	Initialization not done
SHIP_STAT_UNREC_HOST	Unrecognized Host ID
SHIP_STAT_UNREC_FCN	Unrecognized function code
SHIP_STAT_INVLENGTH	Data length was not valid
SHIP_STAT_INVPPSOC	Invalid PP Socket ID words
SHIP_STAT_CMDERROR	Command error on read
SHIP_STAT_CANCELED	Command was canceled



## Read Redirect Noblock Command

This command tells the Protocol Processor that the host wants to read data, and the data should be sent immediately. Data is to be sent to a different host. The data transfer size tells the Protocol Processor the total amount of data that the host wishes to read. If there is no data, the PP returns the response indicating no data available.

The Request ID Ack will tell the PP that all read requests up to but not including this request ID have been received, and the data may be discarded.

Header :

Host ID	Reserved	Function Code = Read Redirect Noblk
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = 0		
Data Length = 0		
Request ID Ack		
Data Transfer Size		

Data:

Address information and header to be used
---

## PP Actions:

If this command exceeds the maximum number of outstanding Read commands, discard the command with no response.

If the Request Id field is the same as the one in the previous Read command, return the same data that was returned by the previous Read command.

Using the Request ID ACK field in this command free the all data awaiting acknowledgment from previous Read commands (i.e. ones with lower Request ID fields).

If there is previously received network data enqueued at the socket

Immediately return a Read Redirect response SHIP packet to the requester, and a Data Packet to the redirected host, containing the data requested or the data available, whichever is smaller, unless the SCB was created with the EXACT DATA SIZE option. If so, and there is less data enqueued than was requested, continue operation as if a Read Redirect (block) was issued.

Enqueue the data buffer at the SCB, and mark the SCB to indicate that Read Data is awaiting acknowledgment.

Otherwise

Immediately return a Read Redirect Response indicating no data available.  
Note: All Read Responses contain the amount of data enqueued for subsequent Reads.

### Read Redirect Noblock Response

This response gives the data to the destination specified in the READ REDIRECT NOBLOCK command. If there is no data, the data length is set to zero.

The server name tells the source from which this data was received. It is only present if there is not a connection already in process. Otherwise, Extension Length is zero.

Header :

Host ID	Status	Function Code = Read Redirect Noblk
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = sizeof (Server Name)		
Data Length = 0		
Command Status		
Reserved		
Server Name		

Data :

Header specified from command
Data being transferred

### Possible Status Values:

SHIP_STAT_MUSTINIT	Initialization not done
SHIP_STAT_UNREC_HOST	Unrecognized Host ID
SHIP_STAT_UNREC_FCN	Unrecognized function code
SHIP_STAT_INVLENGTH	Data length was not valid
SHIP_STAT_INVPPSOC	Invalid PP Socket ID words
SHIP_STAT_CMDERROR	Command error on read

## Peek Command

This command tells the Protocol Processor that the host wants to read data, but do not delete the data when completed. The data length tells the Protocol Processor the total amount of data that the host wishes to peek. If there is no data present, the data length will be zero. Total data requested cannot exceed 992 bytes.

If a read request of any type follows this command, it cannot be redone, as the data will have been sent to the host.

Header :

Host ID	Reserved	Function Code = Peek
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = 0		
Data Length = 0		
Reserved		
Data Transfer Size		

## PP Actions:

If there is previously received network data enqueued at the socket

Immediately return a Peek Response SHIP packet, containing the data requested, up to 1k bytes.

Otherwise

Immediately return a Peek response SHIP packet indicating no data available.

## Peek Response

This response gives the data to the host, but does not delete it from the PP buffers. Data length will specify the amount of data being returned. If no data is available, a zero will be returned.

The server name tells the source from which this data was received. It is only present if there is not a connection already in process. Otherwise, Extension Length is zero.

Header:

Host ID	Status	Function Code = Peek
Request_id		
First Host Socket ID word		
Second Host Socket ID word		
First PP Socket ID word		
Second PP Socket ID word		
Extension Length = sizeof (Server Name)		
Data Length = 0		
Command Status		
Reserved		
Server Name		

Data:

Data being Transferred
------------------------

## Possible Status Values:

SHIP_STAT_MUSTINIT	Initialization not done
SHIP_STAT_UNREC_HOST	Unrecognized Host ID
SHIP_STAT_UNREC_FCN	Unrecognized function code
SHIP_STAT_INVLENGTH	Data length was not valid
SHIP_STAT_INVPPSOC	Invalid PP Socket ID words
SHIP_STAT_CMDERROR	Command error on peek

## **Descriptions**

### **Sequences**

All transactions are sequential. The PP must keep the status of any request (or be able to recreate the status) in case the host needs to redo the transaction. Upon receiving a new transaction with a different request ID, the PP assumes that the previous transaction has been successfully completed and will not be repeated, and thus can discard the status of the last request. Also, the host cannot issue another transaction until it has received a response from the last one, or it has timed out.

The only exception to this is during the READ and WRITE sequences. Multiple READ and WRITE commands may be done simultaneously. Each READ command acknowledged one or more previous READ commands, allowing the PP to discard data that has been successfully transferred to the host.

Multiple WRITE commands may also be done. However, if a WRITE command fails, all subsequent WRITE commands will be discarded with a response of OUT OF SEQUENCE until the failing write is reissued successfully. This prevents out of order data from a successful WRITE command getting out before the data from a previous unsuccessful WRITE command.

### **Errors**

If there is a transmission error on a received message (for example, bad parity on a HIPPI packet), the message is discarded.

If the host fails to receive a response in the specified timeout period, it should reissue the request with the same request ID. If the response was already sent but lost, the Protocol Processor will resend the response.

Blocking requests can be terminated before the response comes by using the CANCEL command. The request will be terminated, and the host will receive a response with a status of CANCELED. If the request finishes before the CANCEL command takes effect, then the host will receive the normal command response, as well as a CANCEL response indicating nothing was available to cancel.

When a connection is terminated, the Protocol Processor will close down the socket and remember the state and error that caused the connection to terminate. The next message sent by the host on that socket will receive a response indicating an error that the socket has been closed. State of the socket will be maintained until the host sends a close or shutdown request.

### **Write Sequence**

To perform a write, the host issues a Socket Write request message telling the Protocol Processor (PP) the server name for the write and the total size of the data to be written. The data to be written is also included in the packet. The PP responds with a Write Response, and processes the data.

## **Read Sequence**

To perform a read, the host issues a Socket Read request message. The PP immediately responds with a Read response, including the data that is currently available, up to the size requested.

If the host detects an error in receiving the data, it can reissue the Read request. The Read request specifies a previous read which it acknowledges the data transferred to the host, and the PP is free to discard those buffers.

The difference between blocking and non-blocking Read requests are when the PP does not currently have any data available. In a non-blocking Read request, the PP returns a status indicating that no data is available. In a blocking Read request, the PP waits for data before returning the response.

## **Read Peek Sequence**

To perform a peek, the host issues a Socket Read Peek request message, telling the PP how much data it wants to receive on this socket, limited to a maximum size of 992 bytes. The PP responds with a Read Peek response with the actual data. It does not discard the data from its buffers.

## **Incoming data on sockets with no read posted**

If the PP receives data on an existing socket for which the host has yet to post a Read Request, the PP will attempt to buffer up a certain amount of this data (probably 32 KBytes or so, depending on available memory) and will wait for the host to issue the read. This data will be unacknowledged to the sending remote-end, so that the PP can discard this data if it needs the buffer space. It is assumed that the remote end will time out and resend this data until it is acknowledged.

When the host finally issues a Read Request on this socket, the PP will then acknowledge this data to the remote-end. This allows the remote-end to send more data to the host.

## **PP Data Requirements for HIPPI**

When using HIPPI, the Protocol Processor deals with bursts, and maintains transmission on burst boundaries. If a burst is in error, the entire burst is discarded. If data is duplicated from a host, duplicate data is discarded to burst boundaries. Once data is acknowledged, it is discarded on burst boundaries. It is recommended that packets be sent and received in full bursts, especially subsequent reads. If the end of a message does not line up on a burst boundary, or the start of the next message does not, the Protocol Processor will use extra processing to line up the data for transmission and will not operate in an optimal fashion.

It is recommended that, for large data transfers ( $> 512$  bytes), the data start at the beginning of the second burst (using the B bit in the FP header); for small transfers ( $< 512$  bytes), the data can be in the same burst as the SHIP header.



## **Acknowledgments**

This project is the work of the Network Engineering Group of the Computing and Communications Division of Los Alamos National Laboratory, where John Morrison is the group leader. The engineers responsible for the design of the protocol are Richard Thomsen, Craig Idler, Marty Halvorson, and Mitch Sukalski.

Also working on the development of the protocol are Jeff Kravitz of the Thomas J. Watson Research Center of IBM in Yorktown Heights, New York, and Brian Beattie who is contractor for the Supercomputer Division of Intel in Hillsborough, Oregon.

## **Appendix A - SHIP Values**

### **SHIP Values**

Values for the SHIP ULP-ID in the HIPPI-FP header are as follows:

SHIP_ULP_ID_MESSAGES	2	SHIP messages
SHIP_ULP_ID_DATA	3	SHIP data (optional)

The function codes for SHIP messages are as follows:

SHIP_FCN_INITIALIZE	1	Initialize communications
SHIP_FCN_CREATE	2	Create Socket
SHIP_FCN_BIND	3	Bind address to socket
SHIP_FCN_CONNECT	4	Connect to remote socket
SHIP_FCN_LISTEN	5	Listen for connection
SHIP_FCN_ACCEPT	6	Accept connection
SHIP_FCN_ACCEPTNOBLK	7	Accept connection - non-blocking
SHIP_FCN_CLOSE	8	Close socket
SHIP_FCN_SHUTDOWN	9	Shut down socket
SHIP_FCN_STATUS	10	Get status of socket
SHIP_FCN_STATUSNOBLK	11	Get status - non-blocking
SHIP_FCN_CANCEL	12	Cancel status of socket
SHIP_FCN_GETSOPTION	13	Get socket option
SHIP_FCN_SETSOPTION	14	Set socket option
SHIP_FCN_WRITE	20	Write data to socket
SHIP_FCN_READ	30	Read data from socket
SHIP_FCN_READNOBLK	31	Read data - non-blocking
SHIP_FCN_READREDIRECT	32	Read data to another host
SHIP_FCN_READREDIRECTNOBLK	33	Read Redirect data - non-blocking
SHIP_FCN_PEEK	34	Read data - peek and do not delete

The status values for SHIP status are as follows:

SHIP_STAT_SUCCESS	0	Successful transaction
SHIP_STAT_MUSTINIT	1	Initialize not done
SHIP_STAT_UNREC_HOST	2	Unrecognized Host ID
SHIP_STAT_UNREC_FCN	3	Unrecognized function code
SHIP_STAT_INVLENGTH	4	Invalid length
SHIP_STAT_INVPPSOC	5	Invalid PP Socket ID words
SHIP_STAT_INVPARAM	6	Invalid parameter to call
SHIP_STAT_CMDERROR	7	Error in command
SHIP_STAT_SEQUENCE	8	Command out of sequence
SHIP_STAT_CANCELED	9	Command was canceled
SHIP_STAT_DESTUNREACH	10	Destination is not reachable
SHIP_STAT_NOROOM	11	No room for write

The values for the address families are as follows:

SHIP_AF_INET	1	Internet address family
--------------	---	-------------------------

The values for the socket types are as follows:

SHIP_SOCKET_TYPE_STREAM	1	Stream protocol type
SHIP_SOCKET_TYPE_DGRAM	2	Datagram protocol type
SHIP_SOCKET_TYPE_RAW	3	Raw type
SHIP_SOCKET_TYPE_SEQPKT	4	Sequential packets
SHIP_SOCKET_TYPE_RDM	5	Reliable delivered message

The values for the protocol types are as follows:

SHIP_PROT_UNSPECIFIED	1	Unspecified protocol
SHIP_IPPROTO_UDP	2	UDP
SHIP_IPPROTO_TCP	3	TCP
SHIP_IPPROTO_ICMP	4	ICMP
SHIP_IPPROTO_RAW	5	Raw

The values for the CREATE option field are as follows:

SHIP_CREATE_OPT_NONE	0	No options
SHIP_CREATE_OPT_EXACT_SIZE	1	Send exact amount of data

The values for the SHIP socket option levels are as follows:

SHIP_SOL_SOCKET	1	Option applied to socket itself
SHIP_SOL_TCP	2	Option applies to TCP
SHIP_SOL_UDP	3	Option applies to UDP
SHIP_SOL_IP	4	Option applies to IP

The values for the SHIP socket options are as follows:

SHIP_SOPT_IP	1	IP options
SHIP_SOPT_TCP_MAXSEG	2	TCP Maximum segment size
SHIP_SOPT_TCP_NODELAY	3	TCP Do not delay send
SHIP_SOPT_SOCK_BROADCAST	4	Permit sending broadcast
SHIP_SOPT_SOCK_DONTROUTE	5	Just use interface address
SHIP_SOPT_SOCK_ERROR	6	Get error status and clear
SHIP_SOPT_SOCK_KEEPAIVE	7	Keep connections alive
SHIP_SOPT_SOCK_LINGER	8	Linger on close if data present
SHIP_SOPT_SOCK_OOBLINE	9	Leave received OOB data in-line
SHIP_SOPT_SOCK_RCVBUF	10	Receive buffer size
SHIP_SOPT_SOCK_SNDBUF	11	Send buffer size
SHIP_SOPT_SOCK_RCVLOWAT	12	Receive low-water mark
SHIP_SOPT_SOCK_SNDLOWAT	13	Send low-water mark
SHIP_SOPT_SOCK_RCVTIMEO	14	Reveive timeout
SHIP_SOPT_SOCK_SNDTIMEO	15	Send timeout
SHIP_SOPT_SOCK_REUSEADDR	16	Allow local address reuse
SHIP_SOPT_SOCK_TYPE	17	Get socket type

SHIP_SOPT_SOC_USELOOPBACK	18	Bypass hardware when possible
SHIP_SOPT_LOC_NOBLOCK	19	Turn on or off local blocking
SHIP_SOPT_LOC_RETRY	20	Turn on or off local retry on error
SHIP_SOPT_LOC_QACK	21	Turn on or off local quick acks

The values for the SHIP states are as follows:

SHIP_STATE_UNINIT	0	Not initialized
SHIP_STATE_IDLE	1	Idle
SHIP_STATE_CREATED	2	Created
SHIP_STATE_BOUND	4	Bound to address
SHIP_STATE_LISTEN	8	Listening
SHIP_STATE_ACCEPT	0x10	Accepted connection
SHIP_STATE_CONNECTING	0x20	Connection in progress
SHIP_STATE_CONNECTED	0x40	Connected
SHIP_STATE_CLOSING	0x80	Closing in progress
SHIP_STATE_STATUS_READ_PEND	0x100	Status read pending
SHIP_STATE_STATUS_WRITE_PEND	0x200	Status write pending

These are values from errno.h that could be returned by socket calls

SHIP_ERR_EPERM	1	Not owner
SHIP_ERR_EINTR	2	Interrupted system call
SHIP_ERR_EIO	3	I/O error
SHIP_ERR_ENXIO	4	No such device or address
SHIP_ERR_E2BIG	5	Arg list too long
SHIP_ERR_ENOMEM	6	Not enough core
SHIP_ERR_EACCES	7	Permission denied
SHIP_ERR_EFAULT	8	Bad address
SHIP_ERR_EINVAL	9	Invalid argument
SHIP_ERR_ENFILE	10	File table overflow
SHIP_ERR_EMFILE	11	Too many open files
SHIP_ERR_EWOULDBLOCK	12	Operation would block
SHIP_ERR_EINPROGRESS	13	Operation now in progress
SHIP_ERR_EALREADY	14	Operation already in progress
SHIP_ERR_ENOTSOCK	15	Socket operation on non-socket
SHIP_ERR_EDESTADDRREQ	16	Destination address required
SHIP_ERR EMSGSIZE	17	Message too long
SHIP_ERR_EPROTOTYPE	18	Protocol wrong type for socket
SHIP_ERR_ENOPROTOOPT	19	Protocol not available
SHIP_ERR_EPROTONOSUPPORT	20	Protocol not supported
SHIP_ERR_ESOCKTNOSUPPORT	21	Socket type not supported
SHIP_ERR_EOPNOTSUPP	22	Op not supported on socket
SHIP_ERR_EPFNOSUPPORT	23	Prot family not supported
SHIP_ERR_EAFNOSUPPORT	24	Addr family not sup by prot family
SHIP_ERR_EADDRINUSE	25	Address already in use

SHIP_ERR_EADDRNOTAVAIL	26	Can't assign requested address
SHIP_ERR_ENETDOWN	27	Network is down
SHIP_ERR_ENETUNREACH	28	Network is unreachable
SHIP_ERR_ENETRESET	29	Network dropped conn on reset
SHIP_ERR_ECONNABORTED	30	Software caused conn abort
SHIP_ERR_ECONNRESET	31	Connection reset by peer
SHIP_ERR_ENOBUFS	32	No buffer space available
SHIP_ERR_EISCONN	33	Socket is already connected
SHIP_ERR_ENOTCONN	34	Socket is not connected
SHIP_ERR_ESHUTDOWN	35	Can't send after sock shutdown
SHIP_ERR_ETOOMANYREFS	36	Too many ref: can't splice
SHIP_ERR_ETIMEDOUT	37	Connection timed out
SHIP_ERR_ECONNREFUSED	38	Connection refused
SHIP_ERR_EHOSTDOWN	39	Host is down
SHIP_ERR_EHOSTUNREACH	40	No route to host
SHIP_ERR_ENOTEMPTY	41	Directory not empty
SHIP_ERR_EPROCLIM	42	Too many processes
SHIP_ERR_EUSERS	43	Too many users
SHIP_ERR_ESTALE	44	
SHIP_ERR_EREMOTE	45	
SHIP_ERR_ENOLCK	46	LOCK_MAX exceeded
SHIP_ERR_ENOSYS	47	Function not implemented
SHIP_ERR_EPIPE	48	Broken pipe (socket not available)

## **Appendix B - SHIP Structures**

The addresses of hosts are passed to SHIP using the SOCKADDR structure. This structure is defined below, and is based on the NET-2 release of BSD networking.

Length	Family	Port
	Address	
	Zero	
	Zero	

The length field is a byte and specifies the size in bytes of the SOCKADDR structure. For internet addresses, the length is always 16.

The Family is a byte that specifies the type of address, and is specified above in the SHIP Values appendix.

The Port is the port to which this address is to be used, and is a 16-bit number.

The Address is an unsigned long, for the Internet address family, and specifies the Internet address of the host.

The next two fields are unused, and must be zero.

## **Appendix C - SHIP State Machines**

The following state machine definitions describe the protocol handling for SHIP, from the point of view of the PP (Protocol Processor), which is event-driven.

The HOST processor is the master in most cases, since it generates most of the events, based on Socket calls from the Application program.

The following table assumes that all operations succeed (no connection attempts refused, etc.). Error condition handling, such as refused connections, timeouts, lost packets, etc., are better described via pseudocode, as the table would become impossible to read if all error conditions were included.

**SHIP Connection State Table**

States-> Input Events	Idle	Connected	Bound	Listen	Accept	Connecting	Connected	Closing
	0	1	2	3	4	5	6	7
INIT	-> 0	-> 0	-> 0	-> 0	-> 0	-> 0	-> 0	-> 0
CREATE	-> 1	Note 1	Note 1	Note 1	Note 1	Note 1	Note 1	Note 1
BIND	Error	-> 2	Error	Error	Error	Error	Error	Error
CONNECT	Error	-> 5	-> 5	Error	Error	Error	Error	Error
LISTEN	Error	Error	-> 3	Error	Error	Error	Error	Error
ACCEPT	Error	Error	Error	-> 4	Error	Error	Error	Error
SHUTDOWN	Error	-> 0	-> 0	-> 0	-> 0	-> 0	-> 7	-> 0
CLOSE	Error	-> 0	-> 0	-> 0	-> 0	-> 0	-> 0	-> 0
READ	Error	Error	Error	Error	Error	Error	Note 3	Error
PEEK	Error	Error	Error	Error	Error	Error	Note 3	Error
WRITE	Error	Error	Error	Error	Error	Error	Note 3	Error
STATUS	Error	Note 2	Note 2	Note 2	Note 2	Note 2	Note 2	Error
CANCEL	Error	Note 2	Note 2	Note 2	Note 2	Note 2	Note 2	Error
GETSOCKOP	Error	Note 2	Note 2	Note 2	Note 2	Note 2	Note 2	Error
SETSOCKOP	Error	Note 2	Note 2	Note 2	Note 2	Note 2	Note 2	Error
Conn Req	Error	Error	Error	Error	-> 5	Error	Error	Error
Conn Cplt	Error	Error	Error	Error	Error	-> 6	Error	Error
Close Req	Error	Error	Error	Error	Error	-> 7	-> 7	-> 7
Close Cpl	Error	Error	Error	Error	Error	-> 0	-> 0	-> 0

### Connection State Definitions

The IDLE state is the state that unused socket control blocks are initialized to.

The CREATED state indicates that the socket has been created, i.e. a socket control block and a local socket ID have been allocated to correspond to the specified Host ID. No external actions (i.e. Transmission over the network) are triggered by entering this state.

The BOUND state indicates that a BIND operation has been issued. No external actions are triggered by entering this state. LISTEN operations are not allowed until this state has been reached.

This LISTEN state indicates that a LISTEN operation has been issued. No external actions are triggered by entering this state. ACCEPT operations are not allowed until this state has been entered.

The ACCEPT state indicates that the Socket is available for connection attempts from external nodes. No external actions are triggered by entering this state. This state can only be exited by receiving a CLOSE or SHUTDOWN request, or by reception of a connection request from the Network.

The CONNECTING state indicates that a CONNECT or ACCEPT command has been issued and that a connection handshake sequence is in progress. This state can be entered from the CREATED state due to the CONNECT command being received, or it can be entered from the ACCEPT state due to a connection request being received from the Network. This can trigger external transmissions which cause the Transport Protocol to complete the connection.



The CONNECTED state is entered after the Transport Protocol has completed the necessary transactions to complete the connection. This state allows data transfers to take place.

The CLOSING state is entered when one partner of the connection has indicated that it is ready to close the connection. When both ends of the connection complete all data transfer, and the necessary transactions to fully close the Transport session have occurred, the Socket control block is set back to the IDLE state.

## **Input Event Definitions**

### **Input Requests from the Host**

#### **INIT**

The INIT request is used to reinitialize the entire SHIP protocol engine. All outstanding buffers, requests, etc. discarded and all control blocks are reset to their initial state.

#### **CREATE**

The CREATE request is used to create a new socket. The socket create response will return a token that uniquely identifies the new socket. This request will fail if insufficient resources exist to allocate a new socket.

#### **BIND**

The BIND request associates a local address with a socket. In the case of a TCP socket, it specifies a TCP port number. This request will fail if the token does not refer to a valid socket or if the socket is already associated with a local address created by a previous bind request or a connect request.

#### **LISTEN**

The LISTEN request specifies the number of incoming connection requests that a previously bound socket may queue up. This call will fail if the socket is invalid or not bound. It will also fail if the socket has an active connection created with a CONNECT request.

#### **ACCEPT**

The ACCEPT request accepts an incoming request on a socket that has been properly initialized by BIND and LISTEN. This request has two forms, blocking and non-blocking. The blocking form waits for an incoming connection before returning a response to the host. The non-blocking form returns a response immediately if an incoming request was queued. If not, it will indicate no connection. A successful ACCEPT request allocates a new socket for the accepted connection and returns the new token to the host.

## CONNECT

The CONNECT request attempts to make a connection to a remote host. The request must specify a remote TCP host and port. The socket may have been the subject of a BIND request but not a LISTEN request. This request will fail if the socket has not been properly initialized, the remote host refuses the connection, or cannot be reached.

## SHUTDOWN

The SHUTDOWN request gracefully closes a connection. The request may indicate no more data will be sent, no more data will be accepted, or no more data will be accepted or sent.

## CLOSE

The CLOSE request causes the input and output queues to be flushed. Any outstanding reads, writes, or status requests will be completed with a status indicating an aborted request. The socket will be closed. The response to this request will be returned as soon as all outstanding requests have been responded to, and the socket has been marked closed. It will not wait for any response from the remote host.

## STATUS

The STATUS request returns a response containing the status of a socket. There are two forms of this request, blocking and non-blocking. The non-blocking form returns immediately. The blocking form returns when either there is data to read, or that data may be sent. If either are true, the response is returned immediately. If the socket is in the listen state, a status request returns the number of incoming connections currently queued.

## CANCEL

The CANCEL request causes a blocking STATUS request on the specified socket to be answered immediately, or it causes a blocking ACCEPT or blocking READ request on the specified socket to be terminated.

## GETSOCKOPT

The GETSOCKOPT request returns the value(s) of the specified option(s). These options are listed in the UNIX man-pages.

## SETSOCKOPT

The SETSOCKOPT request sets the value(s) of the specified option(s).

## Input Events from the Network

"Conn Req" corresponds to a connection request, from the peer network node, indicating that it wishes to begin the connection handshake sequence.

"Conn Cplt" indicates that the connection handshake sequence has completed, causing the connection to be established.

"Close Req" corresponds to a close request, from the peer network node, indicating that it wishes to begin the close handshake sequence.

"Close Cpl" indicates that the close handshake sequence has completed, or a suitable timeout caused the connection to be closed anyway.

Notes:

Note 1. CREATE is issued to allocate a new Socket; by definition the newly allocated Control Block MUST be in the Idle state.

Note 2. These commands can be issued in any state except IDLE or CLOSING. They have the effect of setting various option and status bits in the SCB, but do not cause changes in the State of the Connection State machine.

Note 3. These commands are legal for these states, and do not change the state of this State machine. However, they do affect the data transfer state machines, as described below.